
Instacart 데이터셋 분석 및 LLM 데이터분석 동향

김찬우

통계학과 석사과정

April 15, 2026

목차

- 데이터셋 및 Competition 목표
- 데이터셋 구성
- 모델 및 Metric
- Feature Engineering
- 실험 결과
- LLM 데이터분석

Section 1

데이터셋 및 Competition 목표

Competition 목표

목표

주어진 user의 이전 주문 정보를 바탕으로, 다음 주문에서 어떤 products들이 재구매될지를 예측

- ▶ 예측 대상: user가 재구매할 products set
- ▶ Q1. 이전 주문 정보들이란?
- ▶ A1. user의 이전 개별 주문에서의 products set, 주문 시각, 주문 간 간격, 재구매 여부 등이 있음
- ▶ Q2. 예측 products set에 들어갈 후보는?
- ▶ A2. user가 주어졌을때 그 user가 이전에 주문했던 모든 products 집합

Section 2

데이터셋 구성

데이터 파일 구성

파일	설명	주요 컬럼	크기
orders.csv	주문 메타데이터	user_id, order_number, eval_set	3.4M × 7
order_prior.csv	과거 주문의 상품 기록	order_id, product_id, reordered	32M × 4
order_train.csv	최종 주문의 상품 기록	order_id, product_id, reordered	1.3M × 4
products.csv	상품 정보	product_id, aisle_id, department_id	50K × 4
aisles.csv / departments.csv	카테고리 정보	aisle, department	134, 21

orders.csv

- ▶ 사용자 주문 기록을 담고 있는 핵심 테이블
- ▶ 각 row는 하나의 order를 의미 (product에 대한 정보는 포함하지 않음)

```
orders[orders['user_id']==5]
```

✓ 0.0s

	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
45	2717275	5	prior	1	3	12	NaN
46	1909121	5	prior	2	0	16	11.0
47	2267326	5	prior	3	3	18	10.0
48	157374	5	prior	4	1	18	19.0
49	2196797	5	train	5	0	11	6.0

prior/train.csv

- ▶ 각 order에 담긴 products 들에 대한 정보
- ▶ reordered 는 이전에 주문한 적 있는 상품이 다시 주문되었음을 의미

```
order_prior.head()
```

✓ 0.0s

	order_id	product_id	add_to_cart_order	reordered
0	2	33120	1	1
1	2	28985	2	1
2	2	9327	3	0
3	2	45918	4	1
4	2	30035	5	0

```
order_train.head()
```

✓ 0.0s

	order_id	product_id	add_to_cart_order	reordered
0	1	49302	1	1
1	1	11109	2	1
2	1	10246	3	0
3	1	49683	4	0
4	1	43633	5	1

merged

- ▶ order id를 기준으로 orders, prior, aisle, department 병합
- ▶ 각 row는 (order_i, product_j) 에 대한 정보들을 포함

```
merged
```

											Python
order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order	product_id	add_to_cart_order	reordered	set_type	
2539329	1	prior	1	2	8	NaN	196	1	0	prior	
2539329	1	prior	1	2	8	NaN	14084	2	0	prior	
2539329	1	prior	1	2	8	NaN	12427	3	0	prior	
2539329	1	prior	1	2	8	NaN	26088	4	0	prior	
2539329	1	prior	1	2	8	NaN	26405	5	0	prior	
...	
272231	206209	train	14	6	14	30.0	40603	4	0	train	
272231	206209	train	14	6	14	30.0	15655	5	0	train	
272231	206209	train	14	6	14	30.0	42606	6	0	train	
272231	206209	train	14	6	14	30.0	37966	7	0	train	
272231	206209	train	14	6	14	30.0	39216	8	1	train	

Section 3

모델 및 Metric

Notation

- ▶ i : User index
- ▶ j : Product index
- ▶ $Y_{ij} \in \{0, 1\}$: Target order에서 $user_i$ 의 $product_j$ 에 대한 reorder 여부
- ▶ U_i : $User_i$ 에 대한 구매이력 Dataset
- ▶ P_j : $Product_j$ 에 대한 구매이력 Dataset
- ▶ UP_{ij} : $User_i \times Product_j$ 에 대한 구매이력 Dataset
- ▶ O_i : $User_i$ 의 target order에서의 메타 정보 (요일, 시각, 시점)
- ▶ $\phi_u, \phi_p, \phi_{up}$: feature maps

Notation

- ▶ $X_i^u = \phi_u(U_i)$: User-level features
- ▶ $X_j^p = \phi_p(P_j)$: Product-level features
- ▶ $X_{ij}^{up} = \phi_{up}(UP_{ij})$: User x Product level features
- ▶ $p_{ij}^* = p^*(U_i, P_j, UP_{ij}, O_i)$: True parameter function
- ▶ f_0 : True parameter function under sufficient condition
- ▶ \hat{g} : XGBoost-estimated function
- ▶ t_i : User_i reorder set 예측 시의 threshold

Model

Distribution

$$Y_{ij} \mid U_i, P_j, UP_{ij}, O_i \sim \text{Bernoulli}(p^*(U_i, P_j, UP_{ij}, O_i))$$

- ▶ 주어진 데이터셋들은 implicit feature들을 포함하여 explicit한 형태로 p_{ij}^* 를 추정하고자 한다.

Feature Mapping

$$X_i^u = \phi_u(U_i), \quad X_j^p = \phi_p(P_j), \quad X_{ij}^{up} = \phi_{up}(UP_{ij})$$

- ▶ Mapping 된 feature들이 U, P, UP 에 대한 충분한 정보를 갖고 있다고 가정한다. ($p_{ij}^* \approx f_0$)

Estimation

$$f_0(X_i^u, X_j^p, X_{ij}^{up}, O_i) = P(Y_{ij} = 1 \mid X_i^u, X_j^p, X_{ij}^{up}, O_i)$$

We estimate f_0 by $\hat{g}(X_i^u, X_j^p, X_{ij}^{up})$ where \hat{g} is obtained using XGBoost

Prediction

Prediction Candidates

$$\mathcal{C}_i = \{j : \text{product } j \text{ was previously purchased by user } i\}$$

- ▶ 예측 후보는 사용자 i 가 이전 주문들에서 한 번 이상 구매했던 Products set

Predicted Set

$$\hat{\mathcal{S}}_i = \{j \in \mathcal{C}_i : \hat{p}_{ij} > t_i\}$$

- ▶ threshold t_i 를 초과한 상품들을 reorder set으로 예측

Evaluation Metric

Answer Set

$$\mathcal{S}_i = \{j : \text{product } j \text{ is reordered in the target order of user } i\}$$

- ▶ Train 데이터셋에서 user i 가 구매한 products 중 reordered =1 인 products 집합

F1 Score (per order)

$$F1_i = \frac{2|\hat{\mathcal{S}}_i \cap \mathcal{S}_i|}{|\hat{\mathcal{S}}_i| + |\mathcal{S}_i|}$$

Final Metric

$$\text{Final F1} = \frac{1}{N} \sum_{i=1}^N F1_i$$

- ▶ N : number of target orders (users)

Section 4

Feature Engineering

Feature Engineering

User-level

- ▶ 재주문 비율 (reorder ratio)
- ▶ 평균 주문 간격
- ▶ 평균 장바구니 크기
- ▶ 사용자 특징 클러스터 (gpt 5.4 기반)

Product-level

- ▶ 총 판매량
- ▶ 평균 장바구니 위치 (add_to_cart_order)
- ▶ one-shot 비율 (한번만 구매수 / 전체 구매수)

User × Product, O_i

- ▶ 유저별 해당 상품 구매 횟수
- ▶ 마지막 구매 이후 경과 일자
- ▶ 평균 장바구니 위치 (user-specific)
- ▶ 구매 주기 (평균 재구매 간격)

Section 5

실험 결과

실험 결과

- ▶ global threshold 0.70 을 사용했을 때, Mean F1 Score = 0.380950
- ▶ Competition 종료로 Test set을 이용할 수 없어서, Train set을 train / valid으로 나눠 실험

	order_id	pred_products	true_products
0	36	[24964, 38293, 44359, 19660, 26629]	[43086, 34497, 48679, 46979, 46620, 19660]
1	38	[33731, 8012, 1940]	[21616]
2	170	[6236, 28092, 5077, 34789, 19953, 35124, 33090...]	[6236, 5077, 40354, 13176, 37766, 18394]
3	631	[21137, 14678, 260, 27086, 5450, 32655, 36011]	[21137, 27344, 42265, 9203, 15842, 13829]
4	844	[33120, 11182, 25890]	[14992, 11182, 31766, 21405, 9387, 18599, 28289]

Section 6

LLM 데이터분석

GPT 5.3

- ▶ 제안된 프로세스를 전체적으로 보았을 때 직접 분석과 크게 차이가 없는 것으로 확인

이 문제에서 가장 중요한 건 학습 샘플의 단위(**unit of analysis**) 를 무엇으로 둘지야.

정답은 거의 확실하게:

학습 단위 = (train 주문, 후보 product)

즉,

각 사용자의 마지막 주문(`eval_set = train`)에 대해

“이 사용자가 과거에 샀던 상품들 중 어떤 상품이 이번에도 다시 주문되었는가?”를 예측하는 형태로 만드는 게 가장 자연스럽다.

왜 이렇게 보냐면:

- `train` 주문에는 실제 구매한 product 목록이 있음
- `prior` 주문에는 과거 행동 이력이 있음
- 따라서 **prior**로 **feature**를 만들고, **train**으로 **label**을 만든다

Gemini 3

1. Target 데이터 구성 (Labeling)

최종 예측 모델은 *****특정 사용자(u)가 특정 상품(p)을 이번에 살 것인가(1) 말 것인가(0)***를 판단하는 이진 분류(Binary Classification) 문제로 정의해야 합니다.**

단계별 처리 과정:

1. **기준 설정:** `orders` 파일에서 `eval_set == 'train'` 인 `order_id` 와 `user_id` 쌍을 추출합니다.
2. **후보군(Candidates) 생성:** 각 사용자별로 과거(`prior`)에 한 번이라도 구매했던 모든 `product_id` 리스트를 만듭니다.
3. **정답 마킹:** * 후보군 중 실제 `order_products__train` 에 존재하는 상품은 **Target = 1**
 - 존재하지 않는 상품은 **Target = 0**
4. **데이터 형태:** 최종적으로 아래와 같은 테이블 형태가 되어야 학습이 가능합니다.

Claude sonnet 4.6

- ▶ Agent 각각에게 EDA, Feature engineering, modeling 페르소나를 줘서 분석

"data-analyst 에이전트로 /home/bosco/project/instacart_agent/data_parquet 디렉터리의 테이블들을 분석해줘. 목표: 각 user가 prior에서 산 적 있는 제품 중 다음 주문에 재주문될 제품 집합 예측. test 라벨이 없으니 order_products_train을 user 단위로 valid/test로 나누고 Mean F1로 평가."

Claude sonnet 4.6

▶ Result

최종 지표

	Test	Valid
Mean F1	0.3617	0.3695
AUC	0.8350	0.8503
최적 threshold	0.565	

단계별 핵심 인사이트

Profiling

- 라벨 보유 train user 131,209명, prior 주문 3,240만 행
- 원 데이터 reorder rate 59.9% (주문 내 제품 기준)
- 누수 없음 확인

Claude sonnet 4.6

▶ 분석 결과

EDA

- 부서별 reorder rate 격차 큼: **dairy/eggs 67%** (최고) vs **personal care 32%** (최저)
- user-product 쌍의 **80%가 1회만 구매** — 구매 빈도가 가장 강한 판별 신호

Preprocessing

- 100K user 샘플링 → (user, product) 후보쌍 **647만 개** 생성
- 후보 공간 기준 positive rate는 **9.76%** (원 데이터 59.9%보다 훨씬 희소 — 이게 F1이 낮아 보이는 이유)
- 재주문 없음(**empty set**) user 6,501명 별도 처리

Feature Engineering (26개 피처 중 핵심 2개)

- `up_order_rate`: user 이력 길이로 정규화한 해당 제품 구매 빈도
- `up_orders_since_last`: 해당 제품을 마지막으로 산 뒤 지난 주문 수 (최상위 중요도, 2위 대비 3배)

Claude sonnet 4.6

▶ 추가적인 최적화 요청 결과

Test Mean F1: 0.3617 → 0.3895 (+0.0278, 직접 하셨던 0.38보다 +0.0095)

방법	Test F1
기존 global threshold 0.565	0.3617
Global threshold 최적화 (0.680)	0.3779
Calibrated Faron (Expected F1 max)	0.3895

핵심: LightGBM 원 확률이 calibration이 안 맞아(평균 0.31 vs 실제 양성 비율 9.75%) Faron 공식의 $E[Y]$ 가 과대 추정 → 사용자당 30+개씩 뽑는 과예측 발생. **Isotonic calibration**을 먼저 태우니 평균 9.3개로 수렴하면서 최적. **None** 예측은 약 3% user.

산출물:

- [analysis/data/test_predictions_optimized.parquet](#) — user별 예측/실제/F1/선택 k
- [analysis/insights.jsonl](#) — evaluation insight 추가

LLM 기반 데이터분석

- ▶ Instacart Competition은 단일 dataset이 아닌 여러개의 데이터셋으로 나뉘어있다는 특징이 있음
→ LLM이 데이터셋들을 연결하고, 모델을 설계하는데 어려움을 겪을 것이라고 추측함
- ▶ 예상과 다르게 데이터 병합, 모델 입출력 설계를 잘 하였음

Questions

1. Instacart 데이터셋과 관련된 분석 reference가 충분해서 가능한 것인가?
2. 더 복잡한 형태의 현실 데이터 분석을 LLM Agent가 잘 설계할 수 있을 것인가?

감사합니다